Open Source Software Engineering

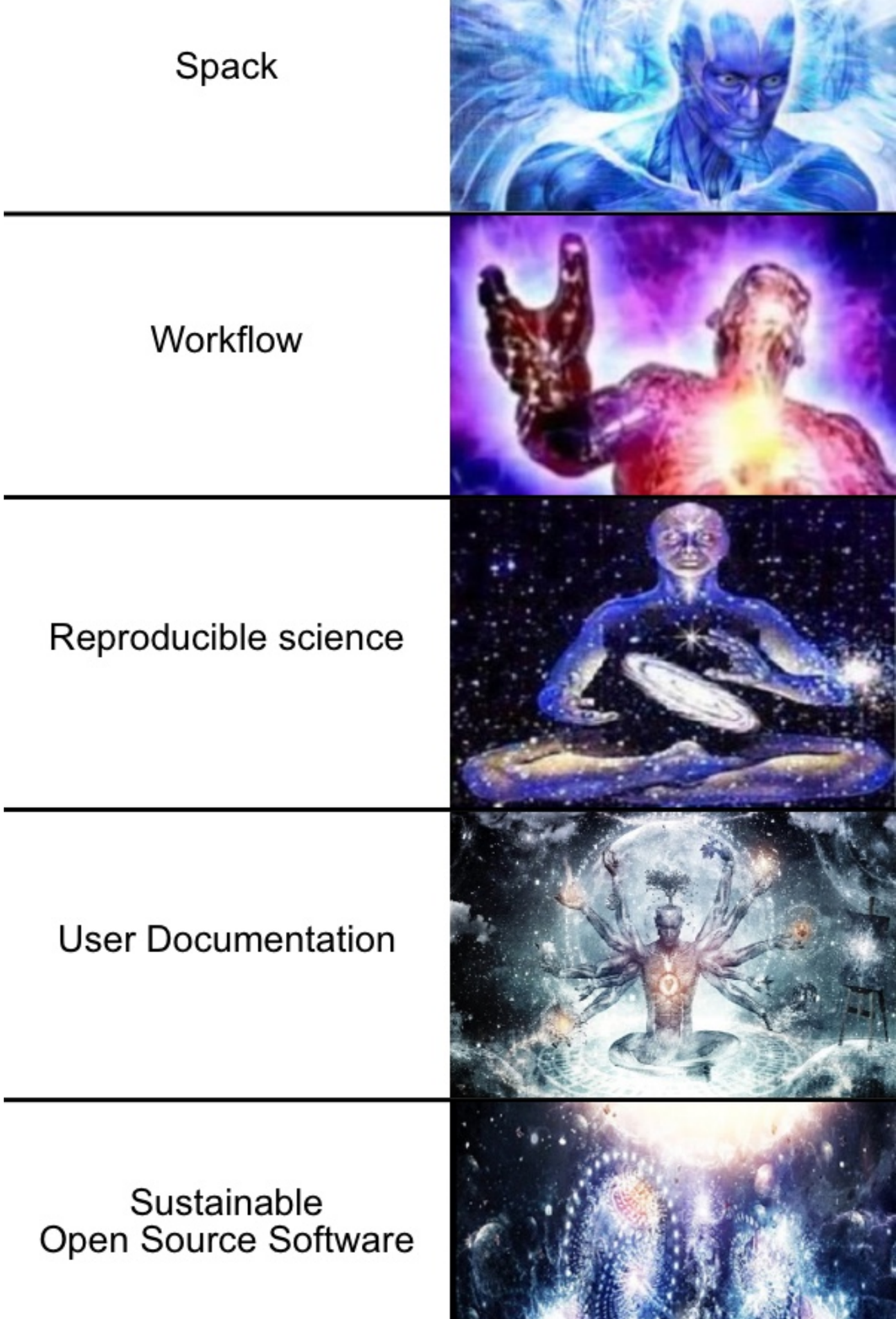
IHPCSS 2018
June 30, 2018
Elsa Gonsiorowski, LLNL, US

Basic

- Unix environment
- git
- GitHub
- CMake (or packaging)
- Doxygen Developer documentation
- Automated Testing

Advanced

- Spack Packaging/Distribution
- Workflow
- ***Reproducible Science***
- User Documentation
- Sustainable Open Source Software



git & GitHub Resources

- Git Immersion: http://gitimmersion.com
- Try Git: https://try.github.io
- GitHub for Education: https://education.github.com https://education.github.com/pack
- ORNL Git Workshop: https://code.ornl.gov/git-it-together/meetings
- http://www.gonsie.com/unix-git/index.html

git – for your advisor

- increased collaboration
- multiple students can contribute simultaneously
- free resources

```
svn up      ⟶  git pull origin master

svn commit  ⟶  git commit -a
               git push origin master
```

Advanced Git

- *Ten Simple Rules for Taking Advantage of git and GitHub:* http://www.biorxiv.org/content/biorxiv/early/2016/04/15/048744.full.pdf

- Git Branching/Cherry pick: http://learngitbranching.js.org

- Undoing, fixing, or removing git commits: https://sethrobertson.github.io/GitFixUm/fixup.html

CMake

- Less brittle
- Easier-to-parse output [doable auto tools: make.org]
- Easier to get help (google)
- cmake.org
  Documentation: https://cmake.org/cmake/help/v3.9/
- Out of source builds => easy to clean up if misconfigured

CMake

```
cmake/
 |- FindJSON.cmake
 `- SetupMPI.cmake
CMakeLists.txt
src/
 |- proj.h
 |- proj.c
 `- CMakeLists.txt
test/
 |- example.c
 `- CMakeLists.txt
doc/
 |- Doxyfile
 `- CMakeLists.txt
```

- Only for packages which aren't in CMake https://github.com/Kitware/CMake/tree/master/Modules
- Follow a template
- See: github.com/LLNL/GOTCHA-tracer

CMake

```
cmake/
 |- FindJSON.cmake
 `- SetupMPI.cmake
CMakeLists.txt
src/
 |- proj.h
 |- proj.c
 `- CMakeLists.txt
test/
 |- example.c
 `- CMakeLists.txt
doc/
 |- Doxyfile
 `- CMakeLists.txt
```

- Called using “include"

CMake

```
cmake/
 |- FindJSON.cmake
 `- SetupMPI.cmake
CMakeLists.txt
src/
 |- proj.h
 |- proj.c
 `- CMakeLists.txt
test/
 |- example.c
 `- CMakeLists.txt
doc/
 |- Doxyfile
 `- CMakeLists.txt
```

- Top-level CMake file
  - setup the project
  - set version number
  - find/import dependencies
  - manage configure-time options

CMake

```
cmake/
 |- FindJSON.cmake
 `- SetupMPI.cmake
CMakeLists.txt
src/
 |- proj.h
 |- proj.c
 `- CMakeLists.txt
test/
 |- example.c
 `- CMakeLists.txt
doc/
 |- Doxyfile
 `- CMakeLists.txt
```

- Manage your code
  - install headers
  - enumerate source files
  - build executables
  - build static and shared libraries

CMake

```
cmake/
 |- FindJSON.cmake
 `- SetupMPI.cmake
CMakeLists.txt
src/
 |- proj.h
 |- proj.c
 `- CMakeLists.txt
test/
 |- example.c
 `- CMakeLists.txt
doc/
 |- Doxyfile
 `- CMakeLists.txt
```

- Testing is important
- Test various modes of using your application
- Compare to known solutions
- Verbose output

CMake

```
cmake/
 |- FindJSON.cmake
 `- SetupMPI.cmake
CMakeLists.txt
src/
 |- proj.h
 |- proj.c
 `- CMakeLists.txt
test/
 |- example.c
 `- CMakeLists.txt
doc/
 |- Doxyfile
 `- CMakeLists.txt
```

- Automatically build documentation for current version of the code

CMake – for your advisor

- Manage special hardware configurations
- Manage complex builds
- Increased student contribution
- Modern!
- Easy to add documentation generation and automated testing

CMake Resources

- Steal from projects that work & look nice [GitHub.com]

- My code: github.com/LLNL/SCR
  From auto tools to CMake in 2 PRs:
  https://github.com/LLNL/scr/pull/41
  https://github.com/LLNL/scr/pull/43

- CMake native packages: https://github.com/Kitware/CMake/tree/master/Modules

- https://github.com/robertmaynard/Sandbox => CMake…

Documentation

- Developer vs. User
- In-source documentation
- Getting started guide?
- Supported Settings
- Advertising

Doxygen

- Can be used on *ANY* code
- Automatically generate caller and callee graphs
- Easy integration with CMake / Travis / GitHub Pages

Function Documentation

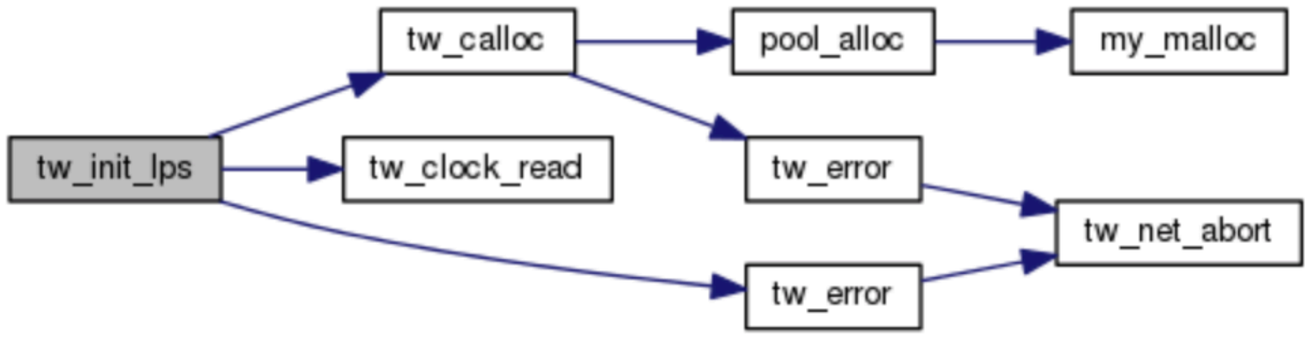
void tw_init_lps (tw_pe * me)

Definition at line **83** of file **tw-lp.c**.

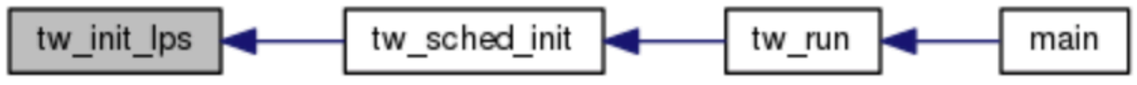
References **tw_pe::abort_event**, **tw_event::caused_by_me**, **tw_pe::cev_abort**, **tw_pe::cur_event**, **tw_lp::cur_state**, **tw_lp::event_counters**, **g_tw_lp**, **g_tw_nlp**, **tw_lptype::init**, **tw_lp::pe**, **tw_lp::prev_event_counters_gvt**, **tw_lp::prev_event_counters_rt**, **tw_lptype::state_sz**, **tw_pe::stats**, **tw_calloc()**, **tw_clock_read()**, **tw_error()**, **TW_LOC**, and **tw_lp::type**.

Referenced by **tw_sched_init()**.

Here is the call graph for this function:



Here is the caller graph for this function:



User Documentation

- Go to the users
- Make it easy to find / read / navigate / get started
- Read the Docs: http://readthedocs.org
- Project Website: https://pages.github.com
- Sphinx: https://www.sphinx-doc.org

Testing

- Test Driven Development TDD (also Behavior DD)
    - Unit Tests
- Automated Tests (Integration tests)
    - Travis CI (continuous integration)
    - Code Coverage
- Regression / Performance Tests

Spack

- Package manager specifically for HPC (and science!)
- Make your software easily available
- Enumerate dependencies
- Contribute to a large HPC OSS project
- github.com/llnl/spack

Additional Tools

- FISh: Friendly Interactive Shell [fishshell.com]
- Sublime Text [sublimetext.com]
- Dot Files [github.com/gonsie/dotfiles]
- GitHub Pages for FREE project website hosting [pages.github.com]
  - Markdown (for your readme files)
  - Jekyll Static Site Generation [jekyllrb.com]

Do Science
Be Social